Phase 2 - Report

Members (Introduction):
Mimi Nguyen - ER Diagrams, Queries (Postgres), Backend (Express.js, Node.js)
Bryan Nguyen - Finalized ER Diagram, Upload/Finalized Files
Karson Arrants - Create Trace File, Recorded/Edited Project
Khanh Nguyen - Frontend (EJS, HTML, Bootstrap), Queries (Postgres), Backend

Application Intro:
This project is a multi-restaurant food order web and database application that is a project made for COSC 3380 Database Systems with Dr. Carlos Ordonez in Fall 2023.

Application Demo: https://youtu.be/YtH1s6OapYw

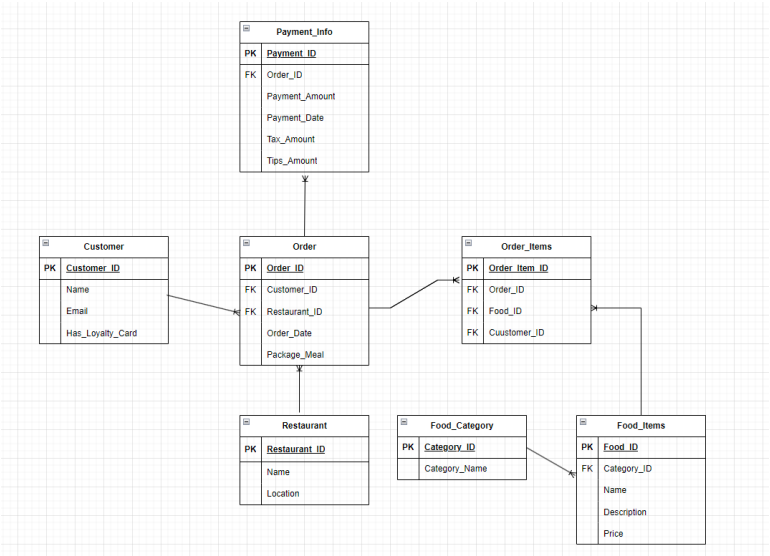
Refer to the readme for project installation and setup:
https://github.com/sixthsenseriot/bites-n-bytes/tree/main

ER Diagram:
https://drive.google.com/file/d/1Zfaqb2Z_8XmRVFczf1OEQFVjklhz7are/view?usp=sharing

Websites Used:
https://www.postgresqltutorial.com/
https://www.postgresql.org/docs/16/index.html
https://getbootstrap.com/docs/4.0/getting-started/introduction/
https://fontawesome.com/

Software Stack:
Database: PostgreSQL
Backend: Express, NodeJS, and EJS
Frontend: EJS and HTML with Bootstrap

DB Model:



Important SQL Queries:

// Pulls data from each selected restaurant from joined tables - Line 21

```
const result = await pool.query(`
    SELECT DISTINCT f.food_id, f.name AS food_name, f.description AS food_description, f.price, c.category_name
    FROM FoodItems f
    JOIN FoodCategories c ON f.category_id = c.category_id
    WHERE f.category_id = $1
`, [restaurantId]);
```

// Pulls data from customer table to render customer info - Line 88

```
const customerResult = await pool.query(`
    SELECT customer_id, name, email, has_loyalty_card
    FROM customer
    WHERE customer_id = $1
`, [customerId]);
```

// Pulls data from joined tables to render customer orders - Line 110

```
const orderResult = await pool.query(`
    SELECT oi.order_item_id, f.name AS food_name, f.price
    FROM OrderItems oi
    JOIN FoodItems f ON oi.food_id = f.food_id
    WHERE oi.order_id IN (SELECT order_id FROM Orders WHERE customer_id = $1)
`, [customerId]);
```

// Adds order to orders table - Line 175

```
const newOrderResult = await pool.query(`
    INSERT INTO Orders (customer_id, restaurant_id, order_date, package_meal)
    VALUES ($1, 1, current_date, false) -- Assuming restaurant ID is 1 and package_meal is set to false
    RETURNING order_id
`, [customerId]);
```

// Deletes order from OrderItems table - Line 207

```
await pool.query(`
    DELETE FROM OrderItems
    WHERE order_item_id = $1
`, [orderItemId]);
```

// Creates CustomerTransaction and OrderHistory tables on place order - Line 296

```
const result = await pool.query(`
CREATE TABLE IF NOT EXISTS CustomerTransaction (
    customer_id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    date DATE NOT NULL,
    transaction VARCHAR(255) NOT NULL
);
`);
```

```
const result = await pool.query(`
CREATE TABLE IF NOT EXISTS OrderHIstory (
    order_item_id SERIAL PRIMARY KEY,
    order_id INT REFERENCES customer(customer_id),
    food_id INT NOT NULL
);
`);
```